A first time primer for reading netcdf files

Summary

This document describes some simple tools that are available for the reading of netCDF files found in the WOCE Global Data set Vol. 2. These tools allow the netCDF files to be read quickly and easily and to be manipulated by many common analysis packages in addition to traditional languages such as Fortran and C/C++ and Java. Because netCDF format is now used for most of the different data products on the WOCE Global Data V2.0, the skills required to read one data set are the same as those needed to read all the others data sets. This reduces the hurdles involved in reading all the different data sets and allows the analyst to concentrate on the interpretation of the observations from many different sources. Because the netCDF files are self-describing, the attributes of each variable (such as units) and name are defined in the data set (along with other information such as the source, creation date, investigator, position, etc etc). This documentation preserves much of the original information in the data set reducing the chance of errors occurring during analysis.

1.0 Why use netCDF for CD ROM 2.0 of WOCE Global Data set?

Much of the WOCE Global Data Resource is now in netCDF format. This data format has many advantages. The most important of which is that it is a self-describing, meaning that software packages can directly read the data and determine its structure, the variables names and essential metadata such as the units. This self-describing aspect of netCDF file format means that the information needed to ensure accurate work (reduce the incidence of errors) is available with the data itself. Secondly, it means that programs describe below can read a netCDF file and generate the code needed to read the file whether it be Fortran, C/C++, JAVA or PERL. Thirdly, plotting and analysis packages (eg FERRET, IDL, MATLAB) can directly read the netCDF files for plotting or analysis.

Although there is an initial learning curve for the inexperienced netCDF user, high efficiency in reading netCDF files and multiple data can be readily achieved as shown in the examples below. These savings translate to more time for analysing data and addressing the synthesis of the WOCE Global Data.

I have assumed that netCDF libraries have already been installed on your system. They can be obtained from www.unidata.ucar.edu/software.html or www.unidata.ucar.edu/netcdf/index.html and are available for Unix, Windows and Macintosh based systems. This site also contains excellent description of the Fortran (also Fortran 90), JAVA and PERL interfaces for netCDF files. Freely available and commercial software that uses netCDF for input are also listed at these sites and includes products such as FERRET, GMT and GrADs and commercial products such as MATLAB, NCAR Graphics, IDL and others. Here we focus on reading netCDF files with FORTRAN, matlab, C, IDL, ncdump and ncbrowse.

2.0 Using FORTRAN to read neCDF files

The simplest way to create the Fortran code you need to read a netCDF is to use the fortran programs cdf2fortran.f (at the above site or http://www-c4.ucsd.edu/~cids/software/visual.html) or gennet.for (www.unidata.ucar.edu/packages/netcdf/contrib.html and http://www.coaps.fsu.edu/WOCE/html/wcdtools.htm). These programs read a netCDF file and create another fortran program with the required calls to the netCDF libraries. This machine generated program only needs to be compiled (with the correct include file and netCDF libraries). When executed this code fragment can then read the netCDF file and is ready for the fortran code to be added to undertake the analysis required by the user. The variable names reflect the internal structure of the netCDF file and this machine generated code is quite readable by humans.

2.1 Worked example of cdf2fortran

The cdf2fortran.f code has been tested on the samples of the netCDF files for the WOCE volume. Retrieve cdf2fortran tar file and use the make file to compile the code. You will have to check that the include directory and the LIBS directory are correct for your particular installation of the netCDF libraries. To compile cdf2fortran just type

```
> make cdf2fortran
```

To execute the cdf2fortran code

```
> cdf2fortran rcm00683.cdf

Generated fortran program called readnet.f

>
```

Either edit readnet.f file and correct the "include ‘netcdf.inc’" file for your particular installation and compile

```
> f77 readnet.f
```

or compile using the -I option to get the correct include file and the -l option to get the correct library files

```
> f77 -c readnet.f -I/usr/local/netcdf/include
> f77 -o readnet readnet.o -l/usr/local/netcdf/lib/libnetcdf.a
```

Sometimes there is a problem at compilation because the variable names may have the same name as a parameter name. Changing the parameter name fixes this problem and now you should have a complete fortran code able to read your particular netCDF file. The result of compiling the above for a sample of the curent meter data on WOCE Global Data CD ROM2 is shown in Appendix 1. Although this piece of code is quite long (compared to the examples shown below) it is reliable and is quick to develop compared reading files that are not self-describing (eg plain ascii files). The sample code fragment can in principle be generalised to read the netCDF data for all of the different current meter data. The program gennet.for works in a very similar way and produces a slightly better documented fortran code. See Appendix 4 for fixing a bug in the readnet.for programs created by gennet.for.

3.0 Using MATLAB to read netCDF files

Commonwealth Scientific Industrial Research Organisation (CSIRO) distributes an interface for the reading of netCDF files (http://www.marine.csiro.au/sw/matlab-netcdf.html). This is a very powerful interface for those who are only interested in reading netCDF files. For those who want to create netCDF files from within matlab then you should use Chuck Denhams interface (http://crusty.er.usgs.gov/~cdenham). You will also need this software to use the CSIRO matlab/netCDF anyway. Follow the install instructions at the above two web sites.

3.1 Sample commands of CSIRO matlab/netCDF interface

Of the 6 main commands, the two most important commands are getnc (getcdf) and inqnc (inqcdf). The detailed instructions on their use is given at the web site (http://www.marine.csiro.au/sw/matlab-netcdf.html) and only the bare bones description is given here. The use of inqnc allows the interactive interrogation of the netCDF file. In matlab, for example:

```
>> inqnc('rcm01037')

--- Global attributes ---

experiment_name: Deep Currents (PCM6)

mooring_name: WHOI 947

pi_name: B.Owens/B.Warren

instrument_type: VACM

latitude: 40.6000

longitude: 147.3492

instrument_depth: 1999.0 m

seafloor_depth: 5280 m

sampling_interval: 30 min

earliest_start_time: 12-jun-1993 12:15:00

latest_stop_time: 01-jul-1995 12:45:00

null_value: -999.9

The 1 dimensions are 1) time = 35954.
```

```
time is unlimited in length

----- Get further information about the following variables -----

-1) None of them (no further information)

0) All of the variables

1) date 2) time 3) speed

4) direction 5) u 6) v

7) temperature 8) pressure

>>
```

creates a description of the contents of the file 'rcm01037'. The global attributes are the information about the data, in this example includes the principal investigator, the instrument name, experiment name etc etc. In addition the routine gives a list of the variables that are contained within the file, in this case there are 8 different variables (date, time, speed etc). In the case of this file the data are organised as vectors with each element corresponding to time and all vectors have a length of 35954. Because 'inqnc' is interactive it is possible to determine the attributes of each of the variables. To retrieve one of the variables within the above data set, such as speed, the getnc command is used.

```
>> speed=getnc('rcm01037','speed');
```

This is the simplest use of the getnc command. Because only two arguments are provided getnc assumes that the data are to be automatically scaled, flagged data will be changed to NaN (not a number), and all data are required. The matlab script in Appendix 2 shows how all of the matlab data in the above test file can be read with just 7 lines of code. Considerably less than the FORTRAN example that reads a slightly different netcdf file but has exactly the same variables shown in Appendix 1. However the development time for the fortran and matlab examples is similar and quick.

The getnc command also has arguments that allows changing the missing values, the stride length (ie allows sub-sampling of the variable) etc. In this way the user can sub-sample, scale the data and read only the variables that are required from the netCDF file. In very large files this represents an important advantage to the user. A nice example of this capability is shown on the above website (by Jim Mansbridge), and also by using the matlab command help where all of the arguments are listed.

```
>> help inqnc
```

All of the matlab scripts, designed to read the netCDF files that appear in the WOCE Global Data V2 are based on just these two matlab commands. The use of the simplest form of getnc means that most of the netCDF files provided from a single WOCE Data Assembly Centre will be readable with a single script. It means that it is very easy and quick to create macros that can read the different netCDF files from the WOCE Data Assembly Centre, leading to significant savings in time and effort. Other important commands are whatnc (gives a list of .nc (.cdf) files in local directory) and attnc (gets the attributes of particular variables like scale).

If necessary the scale factor applied to data can be obtained from the netCDF file by using the attnc comand. For example, in matlab

>>Latitude=getnc(file,'Latitude')*attnc(file,'Latitude','Scale');

would force the Latitude to be correctly scaled by the Scale factor from within the netcCDF file. Only the wind data needs this treatment and is shown the matlab scripts provided with the WOCE Global Data CD Rom.

4.0 Using C to read netCDF files

The simplest way to create the C code you need to read a netCDF is to use the C programs cdf2c.c (at the above site or http://www-c4.ucsd.edu/~cids/software/visual.html). This program reads a netCDF file and creates another C program with the required calls to the netCDF libraries. This machine generated program only needs to be compiled (with correct include file and netCDF libraries). When executed this code fragment can then read the netCDF file and is ready for the C code to be added to undertake the analysis required by the user. The procedure for installing and compiling is described in the documentation from the web site listed above and is similar in operation to the cdf2fortran program described above.

5.0 Using IDL to read netCDF files

IDL can read netCDF files and specifically has a browser to do this (see page 137 of What's new in IDL 5.3). In addition IDL has specific routines for reading netCDF files. However, the simplest way to create the IDL code you need to read a netCDF file is to use the IDL programs cdf2idl (at the above site or http://www-c4.ucsd.edu/~cids/software/visual.html). The concept of this program is similar cdf2fortran.f, cdf2c.c, and cdf2asc.c programs and is provided by the same group (C4 Interactive Display System). This program reads a netCDF file and create another IDL script which can read the netCDF file.

5.1 Example of using cdf2idl

From within IDL all you need to do, to read the current meter data file shown for the cdf2fortran example (Section 2.1) is

```
IDL> .run cdf2idl.pro

IDL> cdf2idl, 'rcm00683.cdf'

IDL> @script.idl
```

At the completion of this script all of the current meter data are in memory ready to be analysed. A total of 3 lines of IDL code needed to read this data set. These three lines of code are so general that they will actually read any of the netCDF files from any of the WOCE Data Assembly Centres provided as part of the WOCE Global Data V2.0.

6.0 Other methods and examining netCDF files

6.1 ncdump

This utility is a very useful program for those who do not have matlab or IDL and want to check the attributes and obtain the data from a netCDF file. On unix machines the man pages describe this command in detail (and its partner ncgen) or in the netCDF User's Guide for Fortran, (Chapter 10). To obtain the attributes of the current meter data one would

```
>ncdump -h rcm00683.cdf
```

Appendix 3 shows the output from this command. To obtain the speed data in this file

```
>ncdump -v speed rcm00683.cdf
```

or to obtain the speed and the direction data

```
>ncdump -v speed, direction rcm00683.cdf
```

The last command can lead to large files ascii files, but have a relatively simple structure and can be read by programs. However, the value of ncdump is that it allows one to check attributes and structure of netcdf files quickly and easily. A more convenient way to examine netCDF files is too be able to visualise the data using a graphical interface such as ncbrowse.

6.2 ncbrowse

This command is a Java application and can be obtained at (http://www.epic.noaa.gov/java/ncBrowse/). This command allows the contents of variables, global attributes to be viewed graphically. It can run on UNIX, Windows and Macintosh's which have the Java Virtual Machine installed (see the above web site or http://java.sun.com/products/jdk/1.2) . On my old PC ncbrowse can run fairly slowly and can be slightly fustrating to uses. However, the interface is largely self explanatory and provides wonderful method for browsing generic netCDF files. Well worth installing and using to check netCDF files as part of the WOCE Global Data V2. Some of the netCDF files on the WOCE Global Data CD Roms are EPIC compliant, which makes the visualisation and plotting of these compliant data even easier.

7.0 Important Web sites

http://www.unidata.ucar.edu/software.htmlUnidata Site (netCDF Libraries and documentation).

http://www-c4.ucsd.edu/~cids/software/visual.htmlC4Interactive Display System site (the suite of programs for reading arbitrary netCDF files and generating fortran, C and IDL code to read the netCDF files.

www.unidata.ucar.edu/packages/netcdf/contrib.html or www.coaps.fsu.edu/WOCE/html/wcdtools.htm gennet.for program

http://www.marine.csiro.au/sw/matlab-netcdf.htmlCSIRO matlab/netCDF interface

http://crusty.er.usgs.gov/~cdenhamChuck Denhams Matlab/netCDF interface (needed with CSIRO matlab/netCDF interface)

http://www.epic.noaa.gov/java/ncBrowse/ncbrowse web site.

Appendix 1: Readnet.for created from cdf2fortran

Sample file produced by the cdf2fortran utility with the netCDF file 'rcm00683.cdf' found in the WOCE CD ROM 2. Note that this fortran routine can be easily generalised to read all of the cm data records on this CD set.

```
c-----------------------------------------------------------------------
c
```

```
c readnet.f

c This file is a fortran template file designed to read the given

c netCDF file into memory.

c

c History:

c Date Name Action

c --------- ------------ --------------------------------------------

c ?? ??? ?? cids Created.

c

c----------------------------------------------------------------------

c Do not forget to include the -I path_to_netcdf_includes in your

c compile statement

c Required includes.

c include 'netcdf.inc'

include '/usr/local/netcdf/include/netcdf.inc'

c Define Variables.

c Variable ids run sequentially from 1 to nvars = 08

parameter (nvars = 8) ! number of variables

parameter (nrec= 36434) ! change this 'to generalize

parameter (ndims = 1) ! number of dimensions

parameter (itime = 36434)

integer*4 rcode ! error code

integer*4 recdim ! record dimension

real*8 date(nrec)

real*8 time(nrec)

real*4 speed(nrec)

real*4 direction(nrec)

real*4 u(nrec)

real*4 v(nrec)

real*4 temperature(nrec)

real*4 pressure(nrec)

integer*4 start(ndims) ! hyperslab starting index

integer*4 count(ndims) ! hyperslab count from start

integer vdims(ndims) ! max # of var dims
```

```
character*1024 strbuf ! string buffer for var

! and attr names

c Open netCDF file.

ncid=ncopn('rcm00683.cdf',ncnowrit,rcode)

c Get info on the record dimension for this file.

call ncinq(ncid,ndims,nvars,ngatts,recdim,rcode)

call ncdinq(ncid,recdim,strbuf,nrecs,rcode)

c nrecs now contains the # of records for this file

c Retrieve data for date variable.

call ncvinq(ncid, 1,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 1,start,count,date,rcode)

c Retrieve data for time variable.

call ncvinq(ncid, 2,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 2,start,count,time,rcode)

c Retrieve data for speed variable.

call ncvinq(ncid, 3,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1
```

```
count(j)=ndsize

end do

call ncvgt(ncid, 3,start,count,speed,rcode)

c Retrieve data for direction variable.

call ncvinq(ncid, 4,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 4,start,count,direction,rcode)

c Retrieve data for u variable.

call ncvinq(ncid, 5,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 5,start,count,u,rcode)

c Retrieve data for v variable.

call ncvinq(ncid, 6,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 6,start,count,v,rcode)

c Retrieve data for temperature variable.

call ncvinq(ncid, 7,strbuf,nctype,nvdim,vdims,nvatts,rcode)
```

```
lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 7,start,count,temperature,rcode)

c Retrieve data for pressure variable.

call ncvinq(ncid, 8,strbuf,nctype,nvdim,vdims,nvatts,rcode)

lenstr=1

do j=1,nvdim

call ncdinq(ncid,vdims(j),strbuf,ndsize,rcode)

lenstr=lenstr*ndsize

start(j)=1

count(j)=ndsize

end do

call ncvgt(ncid, 8,start,count,pressure,rcode)

c ******************************************

c Begin writing statements to use the data.

c ******************************************

c End Program.

stop

end
```

Appendix 2: Example matlab script for current meter data

A matlab macro to read the netCDF file 'rcm01037.cdf' found in the WOCE CD ROM 2. This file is the equivalent of the Fortran code listed in Appendix 1. Note that this macro routine can be easily generalised to read all of the current meter data records on this CD set.

```
%

% A simple Matlab script to read the RCM netcdf file.

%

% Author: Nathan Bindoff

% Date: 28 March 2000

% The script uses Jim Mansbridge's netcdf routines for matlab.

%
```

```
% Documetation and description of these routines can be found at

% http://www.marine.csiro.au/sw/matlab-netcdf.html.

%

%

% Read the current meter data

%

inqnc('rcm01037')

%

%

file='rcm01037'

%

%

date=getcdf(file,'time');

time=getcdf(file,'time');

speed=getcdf(file,'speed');

direction=getcdf(file,'direction');

u=getcdf(file,'u');

v=getcdf(file,'v');

temperature=getcdf(file,'temperature');

pressure=getcdf(file,'pressure');

%

% Example of contents of inqnc('rcm01037')

%

%

% --- Global attributes ---

%experiment_name: Deep Currents (PCM6)

%mooring_name: WHOI 947

%pi_name: B.Owens/B.Warren

%instrument_type: VACM

%latitude: 40.6000

%longitude: 147.3492

%instrument_depth: 1999.0 m

%seafloor_depth: 5280 m

%sampling_interval: 30 min

%earliest_start_time: 12-jun-1993 12:15:00
```

```
%latest_stop_time: 01-jul-1995 12:45:00

%null_value: -999.9

%

%The 1 dimensions are 1) time = 35954.

%time is unlimited in length

%

%----- Get further information about the following variables -----

%

% -1) None of them (no further information)

% 0) All of the variables

% 1) date 2) time 3) speed

% 4) direction 5) u 6) v

% 7) temperature 8) pressure
```

Appendix 3: Example output from ncdump

The attributes of the file rcm00683.cdf obtainded from ncdump -h. This is equivalent to the matlab inqnc command. It describes the global attributes, and each of the variables.

```
netcdf rcm00683 {

dimensions:

time = UNLIMITED ; // (36434 currently)

variables:

double date(time) ;

date:long_name = "date (yr, mo, day)" ;

date:units = "YYYYMMDD" ;

date:valid_range = 19800101, 20101231 ;

double time(time) ;

time:long_name = "time of day" ;

time:units = "hhmmss.dd" ;

time:valid_range = 0., 2359.59 ;

float speed(time) ;

speed:units = "m/sec" ;

speed:valid_range = 0., 3. ;

float direction(time) ;

direction:long_name = "true direction (toward)" ;

direction:units = "degrees" ;
direction:valid_range = 0., 360. ;

float u(time) ;

u:long_name = "eastward velocity component" ;

u:units = "m/sec" ;

u:valid_range = -3., 3. ;

float v(time) ;

v:long_name = "northward velocity component" ;

v:units = "m/sec" ;

v:valid_range = -3., 3. ;

float temperature(time) ;

temperature:long_name = "water temperature" ;

temperature:units = "degrees C" ;

float pressure(time) ;

pressure:units = "decibars" ;

// global attributes:

:experiment_name = "Deep Currents (PCM6)" ;

:mooring_name = "WHOI 942" ;

:pi_name = "B.Owens/B.Warren" ;

:instrument_type = "VACM" ;

:latitude = " 36.4025" ;

:longitude = " 150.2307" ;

:instrument_depth = "2001.0 m" ;

:seafloor_depth = " 5829 m" ;

:sampling_interval = " 30 min" ;

:earliest_start_time = "08-jun-1993 12:15:00" ;

:latest_stop_time = "07-jul-1995 12:45:00" ;

:null_value = -999.9 ;

}
```

Appendix 4: Bug in gennet.for

The program in gennet.for is not quite perfect. When it creates the readnet.for program it some times places the wrong routine call in readnet.for. If you get a message like

>NCAGT: : Attempt to convert between text & numbers

the following may fix the problem. The line where readnet.for crashes is a call to NCAGT. At this point, readnet.for tries to read a character variable atribute "long_name" using a routine that is designed for numeric attributes (NCAGT). To make the code work substitute a call to NCAGTC at this point in the code and also to assign mtlen=50 (the character width dimension of "name"). With these two changes the code runs like a charm.